

每日分享：

加油！你一定可以！你是最胖的！！！！

-----

-----

js学习：

js的变量

js的运算符和逻辑结构

js的数组

js的函数

js的事件机制

js 的常用方法和对象

js的常用方法和对象：

作用:js将常用的功能已经封装好，调用即可，不用重复的封装了。

-----

-----

```
<html>
```

```
  <head>
```

```
    <title>js的常用方法和对象学习</title>
```

```
    <meta charset="UTF-8"/>
```

```
  <!--
```

js的常用方法和对象学习：

String对象：操作字符的。

使用：字符串.函数名即可

大小写转换：

toUpperCase() 转换大写

toLowerCase() 转换小写

字符串截取

substr(0,1)      从指定开始位置截取指定  
长度的子字符串

substring(0,1)    从指定位置开始到指定的  
结束位置的子字符串（含头不含尾）

查找字符位置

indexOf      返回指定字符第一次出现的位  
置。

lastIndexOf      返回指定字符最后一次出  
现的位置。

Date对象：

使用：var 变量名=new Date();

注意：获取的是客户端的时间，不能作为系统功能  
校验的时间的。

参照：API

Math对象：

使用：Math.函数名

random()

round()

ceil()

floor()

Global对象

eval()

isNaN()

parseInt()

parseFloat()

-->

<script type="text/javascript">

//String对象

function testString(){

//创建字符串

var str="abcdefg";

//大小写转换

alert(str.toUpperCase()+":"+str.toLowerCase());

//字符串截取

alert(str.substr(0,1)+":"+str.substring(0,1));

```
    alert(str.substr(0,1).toUpperCase()+str.substr(1,str.length));  
    }  
    //Date对象  
    function testDate(){  
        //创建日期对象  
        var d=new Date();  
        //操作日期和时间  
        //获取年份  
        alert(d.getFullYear());//返回从1900年起算距今  
的年份数  
        alert(d.getFullYear());//返回当前的年份  
        //获取月份  
        alert(d.getMonth()+1);//获取当前的月份数  
( 注意：要+1 )  
        //获取日期  
        alert(d.getDate());//返回当前的日期  
        //获取星期数  
        alert(d.getDay());//返回星期数，注意星期日  
返回0  
        //获取小时数
```

```
        alert(d.getHours()); //返回当前的小时数
        //获取分钟数
        alert(d.getMinutes()); //返回当前的分钟数
        //获取秒数
        alert(d.getSeconds()); //返回当前的秒数
    }

    //Math对象
    function testMath(){
        //随机数字

        alert(Math.floor(Math.random()*9000+1000)); //可以作
为验证码

    }

    //Global对象
    function testGlobal(){
        //eval方法 将字符串转换为js代码执行
        eval("var a='123';");
        alert(a);

        //isNaN 判断Number强转后是否是数字。
        if(!isNaN(a)){
            alert("是数字")
        }
    }
}
```

```
        }else{
            alert("不是数字")
        }
    }
</script>
</head>
<body>
    <h3>js的常用方法和对象学习</h3>
    <hr />
    <input type="button" id="" value="测试String对象"
onclick="testString()"/>
    <input type="button" id="" value="测试Date对象"
onclick="testDate()"/>
    <input type="button" id="" value="测试Math对象"
onclick="testMath()"/>
    <input type="button" id="" value="测试Global对象"
onclick="testGlobal()"/>

</body>
</html>
```

## js 的 window 对象的常用方法

js的window对象的常用方法：

```
<html>
```

```
<head>
```

```
<title>js的常用方法和对象学习</title>
```

```
<meta charset="UTF-8"/>
```

```
<!--
```

js的常用方法和对象学习：

String对象：操作字符的。

使用：字符串.函数名即可

大小写转换：

toUpperCase() 转换大写

toLowerCase() 转换小写

字符串截取

substr(0,1)      从指定开始位置截取指定  
长度的子字符串

substring(0,1)    从指定位置开始到指定的  
结束位置的子字符串（含头不含尾）

查找字符位置

indexOf      返回指定字符第一次出现的位  
置。

lastIndexOf      返回指定字符最后一次出

现的位置。

Date对象：

使用：var 变量名=new Date();

注意：获取的是客户端的时间，不能作为系统功能  
校验的时间的。

参照：API

Math对象：

使用：Math.函数名

random()

round()

ceil()

floor()

Global对象

eval()

isNaN()

parseInt()

parseFloat()

-->

```
<script type="text/javascript">
```

```
//String对象
```

```
function testString(){
```

```
//创建字符串
```



```
var str="ABCDEFGH";
```

```
//大小写转换
```

```
alert(str.toUpperCase()+" "+str.toLowerCase());
```

```
//字符串截取
```

```
alert(str.substr(0,1)+" "+str.substring(0,1));
```

```
alert(str.substr(0,1).toUpperCase()+str.substr(1, str.length));
```

```
}
```

```
//Date对象
```

```
function testDate(){
```

```
//创建日期对象
```

```
var d=new Date();
```

```
//操作日期和时间
```

```
//获取年份
```

```
alert(d.getFullYear());//返回从1900年起算距今
```

的年分数

```
alert(d.getFullYear());//返回当前的年份
```

```
//获取月份
```

```
alert(d.getMonth()+1);//获取当前的月份数
```

( 注意：要+1 )

```
//获取日期
```

```
alert(d.getDate());//返回当前的日期
```

```
//获取星期数
```

```
alert(d.getDay());//返回星期数，注意星期日
```

返回0

```
//获取小时数
```

```
alert(d.getHours());//返回当前的小时数
```

```
//获取分钟数
```

```
alert(d.getMinutes());//返回当前的分钟数
```

```
//获取秒数
```

```
alert(d.getSeconds());//返回当前的秒数
```

```
}
```

```
//Math对象
```

```
function testMath(){
```

```
    //随机数字
```

```
    alert(Math.floor(Math.random()*9000+1000));//可以作
```

为验证码

```
}
```

```
//Global对象
```

```
function testGlobal(){
    //eval方法 将字符串转换为js代码执行
    eval("var a='123';");
    alert(a);
    //isNaN 判断Number强转后是否是数字。
    if(!isNaN(a)){
        alert("是数字")
    }else{
        alert("不是数字")
    }
}

</script>
</head>
<body>
    <h3>js的常用方法和对象学习</h3>
    <hr />
    <input type="button" id="" value="测试String对象"
onclick="testString()"/>
    <input type="button" id="" value="测试Date对象"
onclick="testDate()"/>
    <input type="button" id="" value="测试Math对象"
onclick="testMath()"/>
```

```
<input type="button" id="" value="测试Global对象"
onclick="testGlobal()"/>
```

```
</body>
</html>
```

## js 的 window 对象的常用属性

js的window对象的常用属性：

```
<html>
  <head>
    <title>window对象的属性学习</title>
    <meta charset="UTF-8"/>
    <!--
```

window对象的属性学习

opener属性

location属性：

作用：地址栏属性，该属性是一个对象，封存了浏览器对地址栏的操作信息

例如：url

使用：

URL操作：

window.location.href//返回当前网页的

## URL信息

```
window.location.href="资源路径"//跳
```

转指定的URL资源。

页面刷新：

作用:重新加载页面资源。

```
window.location.reload();
```

-----

history属性：

forward()//前进，相当于浏览器中的前进功能

back()//后退，相当于浏览器中的后退功能

go()//跳转指定的历史记录

screen属性

window.screen.width

window.screen.height

获取分辨率

-->

```
<script type="text/javascript">
```

```
//location属性
```

```
//URL操作
```

```
function testLocation(){
```

```
    alert(window.location.href);//查看当前网
```

## 页的URL信息

```
//window.location.href="http://www.jd.com";

        window.location.href="02-js的window对象学
习 (1).html"//调转指定的资源

    }

    //页面刷新

    function testRefresh(){

        window.location.reload();

    }

    //history属性

    function testHistory(){

        //前进

        //window.history.forward();//前进,相当
于浏览器中的前进功能

        //后退

        //window.history.back();//后退,相当于
浏览器中的后退功能

        //跳转指定历史记录

        window.history.go(0); //跳转指定的历史
记录

    }
```

```
//Screen属性

function testScreen(){

    alert(window.screen.width+": "+window.screen.height)

;

    }

</script>

</head>

<body>

    <h3>window对象的属性学习</h3>

    <hr />

    <input type="button" id="" value="测试location属性
--URL操作" onclick="testLocation()"/>

    <input type="button" id="" value="测试location属性
--刷新页面" onclick="testRefresh()"/>

    <hr />

    <input type="button" id="" value="测试history"
onclick="testHistory()"/>

    <hr />

    <input type="button" value="测试Screen"
onclick="testScreen()"/>

</body>
```

</html>

## js 的 document 获取 HTML 元素对象

js的document获取HTML元素对象（DOM编程）：

```
<html>
```

```
<head>
```

```
<title>js的document对象学习</title>
```

```
<meta charset="UTF-8"/>
```

```
<!--
```

```
    documnet对象学习：
```

解释：

document对象是浏览器对外提供的封存了当前运行的HTML文档信息的

一个对象

通过此对象可以让我们灵活的操作HTML文档，达到修改，完善HTML文档

的目的。

使用：

document获取HTML元素对象

直接 方式：

通过ID

```
var 变量名
```

```
=document.getElementById("uname");//返回指定的HTML元素对象
```

注意：



此对象中封存了HTML元素标签的所有信息。

通过name

```
var 变量名=document.getElementsByName("name属性值");
```

注意：

获取的是相同name属性的HTML元素对象的数组。

通过标签名

```
var 变量名=document.getElementsByTagName("标签名");
```

注意：

返回的是存储了该网页中指定的标签的所有对象的数组。

间接方式：

父子关系：

先获取父级节点(参照直接方式)

通过父级节点获取子节点数组

```
var 变量名=父节点对象.childNodes;
```

注意：

子节点数组中会包含文本节点，可以使用  
nodeType属性

筛选出来所有的HTML节点(nodeType值为1)

子父关系：

先获取子元素对象(参照直接方式)

通过子元素对象获取父元素对象

```
var 变量名=子元素对象.parentNode
```

兄弟关系：

先获取当前元素

根据兄弟关系选择对应的获取方式

```
var 变量名=元素对象.previousSibling; //兄
```

```
var 变量名=元素对象.nextSibling;    //弟
```

document操作元素对象的属性

document操作元素对象的样式

document操作元素对象的文档结构

-->

```
<script type="text/javascript">
```

```
//document获取HTML元素对象
```

```
//通过ID获取
```

```
function testById(){
```

```
var
```

```
inp=window.document.getElementById("uname");
```

```
    alert(inp.value);
```

```
}
```

```
//通过name获取
```

```
function testByName(){
```

```
var
```

```
fav=document.getElementsByName("fav");

    alert(favs[0].value);

}

//通过标签

function testByTagName(){

    var

inps=document.getElementsByTagName("input");

    alert(inps.length);

}

/*-----

*/

//间接方式

//父子关系

function testGetChild(){

    //先获取父级元素对象

    var

p=document.getElementById("showdiv");

    //获取子元素对象数组

    var clds=p.childNodes;

    //遍历

    var arr=[];
```

```
        for(var i=0;i<clds.length;i++){
            if(clds[i].nodeType==1){
                arr.push(clds[i]);
            }
        }
        alert(arr.length);
    }
//子父关系
function getParent(){
    //获取子元素对象
    var child=document.getElementById("un");
    //获取父元素对象
    var p=child.parentNode;
    alert(p)
}
//兄弟关系
//兄关系
function getPreEle(){
    //获取当前元素对象
    var un=document.getElementById("un");
    //获取兄元素对象
    var inp=un.previousSibling;
```

```
        alert(inp);
    }

    //弟关系

    function getNextEle(){

        //获取当前元素对象

        var un=document.getElementById("un");

        //获取兄元素对象

        var inp=un.nextSibling;

        alert(inp);

    }

</script>

</head>

<body>

    <h3>js的document对象学习</h3>

    <hr />

    <input type="button" id="" value="测试通过ID获取"
onclick="testById()"/>

    <input type="button" id="" value="测试通过name获取"
" onclick="testByName()"/>

    <input type="button" id="" value="测试通过标签名获取"
onclick="testByTagName()"/>

    <hr />
```

```
    用户名:<input type="text" name="" id="uname"
value="张三" /><br />
    爱好:<br />
        <input type="checkbox" name="fav" id=""
value="1" />唱歌<br />
        <input type="checkbox" name="fav" id=""
value="1" />跳舞<br />
        <input type="checkbox" name="fav" id=""
value="1" />旅游<br />
        <input type="checkbox" name="fav" id=""
value="1" />吃鸡<br />
    <hr />
    <input type="button" id="" value="测试父子关系"
onclick="testGetChild()" />
    <input type="button" id="" value="测试子父关系"
onclick="getParent()" />
    <input type="button" id="" value="测试兄关系"
onclick="getPreEle()" />
    <input type="button" id="" value="测试弟关系"
onclick="getNextEle()" />
    <hr />
    <div id="showdiv" style="border: solid 1px;
```

```
background-color: orange; width: 300px; height:
300px;" >
    <input type="text" value="张三" id="un"
style="margin: 10px;"/>
    <input type="text" value="李思思" style="margin:
10px;"/>
    <input type="text" value="王五" style="margin:
10px;"/>
    <input type="text" value="赵信" style="margin:
10px;"/>
    <input type="text" value="慎" style="margin:
10px;"/>
    </div>

</body>
</html>
```

## js 的 document 操作元素属性

js的document操作元素属性：

```
<html>
<head>

    <title>操作元素属性</title>
    <meta charset="UTF-8"/>
```

<!--

获取HTML元素对象

操作元素属性：

获取：

元素对象.属性名//返回属性值

修改：

元素对象.属性名=值

注意：

不要修改标签的name和id属性

-->

<script type="text/javascript">

//获取元素属性

function getFieldData(){

//获取HTML元素对象

var inp=document.getElementById("uname");

//获取元素属性

alert(inp.type+":"+inp.name+":"+inp.id+":"+inp.value);  
}

//修改元素属性

function updateFiledData(){

//获取HTML元素对象

var inp=document.getElementById("uname");

//修改元素属性

inp.type="button";

inp.id="un";

}

</script>



```
</head>
<body>

    <h3>操作元素属性</h3>

    <input type="button" id="" value="操作属性--获取"
onclick="getFieldData()" />

    <input type="button" id="" value="操作属性--修改"
onclick="updateFiledData()" />
    <hr />

    用户名:<input type="text" name="uname" id="uname"
value="zhangsan" />
    </body>
</html>
```

## js 的 document 操作内容

### js的document操作内容：

```
<html>
<head>

    <title>操作元素内容</title>

    <meta charset="UTF-8"/>
    <!--
```

获取HTML元素对象

操作元素的内容

获取：

元素对象.innerHTML

返回所有的内容包括HTML标签

元素对象.innerText

返回所有的文本内容，不包括HTML标签

修改：

元素对象.innerHTML="新的内容"

HTML标签会被解析，覆盖原有内容

元素对象.innerText="新的内容"

HTML标签不会被解析，覆盖原有内容

注意：

如果在修改是需要保留原有内容，则使用：

元素对象名+="新的内容";

-->

<script type="text/javascript">

//操作元素内容

//获取

function getData(){

//获取HTML元素对象

var showdiv=document.getElementById("showdiv");

//获取元素内容

alert(showdiv.innerHTML); //返回所有的内容包括

HTML标签

alert(showdiv.innerText); //返回所有的文本内容，不

包括HTML标签

}

//修改

function updateData(){

//获取HTML元素对象

var showdiv=document.getElementById("showdiv");

//修改元素内容

//showdiv.innerHTML+="<u>你的，去前面探路的干

活</u>";//HTML标签会被解析，覆盖原有内容

showdiv.innerText="<u>你的，去前面探路的干活

</u>";//HTML标签不会被解析，覆盖原有内容

}

</script>

<style type="text/css">

#showdiv{

border: solid 1px;

width: 200px;

height: 200px;

background-color: orange;

}

</style>

</head>

<body>

<h3>操作元素内容</h3>

<input type="button" id="" value="操作元素内容---获取"

onclick="getData()"/>

<input type="button" id="" value="操作元素内容---修改"

onclick="updateData()"/>

<hr />

<div id="showdiv">

<b>皇军，前面有八路的干活儿</b>

</div>

</body>

</html>

## js 的 document 操作元素样式

js的document操作元素样式：

<html>

<head>

```
<title>操作元素样式</title>
```

```
<meta charset="UTF-8"/>
```

```
<!--
```

获取HTML元素对象

操作样式：

添加样式：

元素对象名.style.样式名=样式值

修改样式：

元素对象名.style.现有样式名=新的样式值

注意：

还可以通过元素对象.className="类选择器名"来添加样式。

```
-->
```

```
<script type="text/javascript">
```

```
//操作元素样式
```

```
//添加样式
```

```
function addCss(){
```

```
    //获取元素对象
```

```
    var showdiv=document.getElementById("showdiv");
```

```
    //添加样式
```

```
    showdiv.style.border="solid 1px";
```

```
    showdiv.style.width="200px";
```

```
    showdiv.style.height="200px";
```

```
    showdiv.style.backgroundColor="orange";
```

```
}
```

```
//使用类选择器添加
```

```
function addCssByClass(){
```

```
    //获取元素对象
```

```
    var showdiv=document.getElementById("showdiv");
```

```
    //添加样式
```

```
        showdiv.className="common";
    }
    //修改样式
    function updateCss(){
        //获取元素对象
        var showdiv=document.getElementById("showdiv");
        //修改样式
        showdiv.style.backgroundColor="darkcyan";
    }

    var arr=["red","darkcyan","orange","blue","gray"];
    //闪图
    function testColor(){
        window.setInterval(function(){
            //获取元素对象
            var
showdiv=document.getElementById("showdiv");
            //随机
            var num=Math.floor(Math.random()*5);
            //修改背景色
            showdiv.style.backgroundColor=arr[num]
        },100);
    }

</script>
<style type="text/css">
    .common{

        border: solid 1px;
        width: 200px;
        height: 200px;
        background-color: red;

    }
</style>
</head>
```

```
<body>

  <h3>操作元素样式</h3>

  <input type="button" id="" value="添加样式---style"
onclick="addCss()"/>

  <input type="button" id="" value="添加样式--className"
onclick="addCssByClass()"/>

  <input type="button" id="" value="修改样式"
onclick="updateCss()"/>


  <input type="button" id="" value="开始闪现模式"
onclick="testColor()" />
  <hr />
  <div id="showdiv">


  </div>
</body>
</html>
```

## js 的 document 操作元素文档结构

js的document操作文档结构：

```
<html>
<head>

  <title>操作文档结构</title>

  <meta charset="UTF-8"/>
  <!--
```

文档结构:指的是HTML文档的树形结构。

节点：HTML文档树中的基本元素单位称为一个节点。

获取HTML元素对象

操作文档结构

新建节点:

```
var 变量名=document.createElement("标签名");//返
```

回创建的HTML元素对象

添加节点

```
节点对象.appendChild(节点对象);
```

删除节点

```
节点对象.removeChild(节点对象);
```

```
-->
```

```
<script type="text/javascript">
```

```
//文档操作
```

```
function operDocu(){
```

```
//获取元素对象
```

```
var showdiv=document.getElementById("showdiv");
```

```
//创建节点
```

```
var inp=document.createElement("input");
```

```
inp.type="file";//设置类型
```

```
//创建节点--换行符
```

```
var br=document.createElement("br");
```

```
//创建节点--删除按钮
```

```
var bt=document.createElement("input");
```

```
bt.type="button";
```

```
bt.value="删除";
```

```
bt.onclick=function(){  
    showdiv.removeChild(inp);  
    showdiv.removeChild(bt);  
    showdiv.removeChild(br);  
}
```

```
//将节点添加到指定的节点中
```

```
showdiv.appendChild(inp);
```

```
        showdiv.appendChild(bt);
        showdiv.appendChild(br);
    }
</script>
</head>
<body>

    <h3>操作文档结构</h3>

    <input type="button" id="" value="点击上传"
onclick="operDocu()"/>
    <hr />
    <div id="showdiv" style="border: solid 1px;">

        </div>
</body>
</html>
```